State/event based versus purely Action or State based Logics

James Smith
jecs@imperial.ac.uk

Abstract

Although less studied than purely action or state based logics, state/event based logics are becoming increasingly important. Some systems are best studied using structures with information on both states and transitions, and it is these structures over which state/event based logics are defined. The logic UCTL and its variants are perhaps the most widely studied and implemented of these logics to date. As yet, however, no-one seems to have defined UCTL*, a trivial step but a worthwhile one. Here we do just that, and define mappings that preserve truth between this logic and its more commonplace fragments CTL* and ACTL*. Also, acknowledging the importance of modal transition systems, we define a state/event based logic over a modified modal transition system as a precursor to further work.

The future is to some extent, even if it is only a very small extent, something we can make for ourselves. - Arthur Prior

1 Introduction

We define the logic UCTL* over Kripke transition systems [10], otherwise known as doubly labelled transition systems [3]. As suggested in [2], our Kripke transition systems and labelled transition systems [9] carry sets of actions rather than just a single action on each transition, making mappings between these structures and Kripke structures simpler. We define ACTL* [2] and ACTL[1] over these structures. We make small changes to the action formulae of [2] to support sets of actions, bringing them in line with those of [12].

Inspired by [2], we show some details of the proofs that the mappings between ACTL* and CTL* preserve truth and add mappings from UCTL* to both ACTL* and CTL*. In a similar vein we show some details of the proofs that the mappings between ACTL and CTL preserve truth and add mappings from UCTL [12] to both ACTL and CTL.

We also briefly look at 3-valued logics. We define a variant of Kripke modal transition systems [7] which again carries sets of actions on each transition rather than just a single action. In order to accomodate this change, we replace *must* and *may* transitions [11] with ! and ? modifiers on the actions. We define the 3-valued logic UPML, a variant of 3-valued PML [1] which includes features of 3-valued $\text{PML}^{\texttt{Act}}$ [6].

2 Definitions for 2-valued logics

We define the structures for 2-valued logics, some common concepts, and the syntax and semantics and these logics. In the case of labelled transition systems and Kripke transition systems, note that in the definitions which follow we limit the number of transitions between any two states in any one direction to at most one.

Definition 2.1. *A labelled transition system or LTS is a tuple $(S, Act, \longrightarrow)$ where:*

- *S is a set of states ranged over by $s, s', s_0, s_1, \ldots$,*
- *Act is set of actions ranged over by a with $\alpha, \alpha_0, \alpha_1, \ldots$ ranging over 2^{Act},*
- *$\longrightarrow \subseteq S \times 2^{Act} \times S$ is the transition relation with $(s_0, \alpha, s_1) \in \longrightarrow$,*

[1] There is more than one logic given the name ACTL in the literature. The only one we refer to is the branching time logic for labelled transition systems proposed in [2].

- *For any two transitions,* $(s_0, \alpha_0, s_1), (s_0, \alpha_1, s_1) \in \longrightarrow \Rightarrow \alpha_0 = \alpha_1$.

Definition 2.2. *A Kripke structure or KS is a tuple* $(S, \longrightarrow, AP, \mathcal{L})$ *where:*

- S *is a set of states ranged over by* $s, s', s_0, s_1, ...$,
- $\longrightarrow \subseteq S \times S$ *is the transition relation with* $(s_0, s_1) \in \longrightarrow$,
- AP *is a set of atomic propositions ranged over by* p,
- $\mathcal{L} : S \times AP \longrightarrow \{true, false\}$ *is an interpretation function that associates a value of true or* $false$ *with each* $p \in AP$ *for each* $s \in S$.

Definition 2.3. *A Kripke Transition System or KTS is a tuple* $(S, Act, \longrightarrow, AP, \mathcal{L})$ *where:*

- S *is a set of states ranged over by* $s, s', s_0, s_1, ...$,
- Act *is set of actions ranged over by* a *with* $\alpha, \alpha_0, \alpha_1, ...$ *ranging over* 2^{Act},
- $\longrightarrow \subseteq S \times 2^{Act} \times S$ *is the transition relation with* $(s_0, \alpha, s_1) \in \longrightarrow$,
- AP *is a set of atomic propositions ranged over by* p,
- $\mathcal{L} : S \times AP \longrightarrow \{true, false\}$ *is an interpretation function that associates a value of true or* $false$ *with each* $p \in AP$ *for each* $s \in S$,
- *For any two transitions,* $(s_0, \alpha_0, s_1), (s_0, \alpha_1, s_1) \in \longrightarrow \Rightarrow \alpha_0 = \alpha_1$.

Note that since transitions carry sets of actions and not just one, there is no silent action τ. Instead the empty set $\{\}$ is considered silent.

For convenience we overload $\mathcal{L}$ and define, for each $s \in S$, the function $\mathcal{L}(s) : AP \longrightarrow \{true, false\}$ where $\mathcal{L}(s)(p) = \mathcal{L}(s, p)$. The set $\{\mathcal{L}(s) \mid s \in S\}$ is ranged over by $\omega, \omega_0, \omega_1, ...$ and on occasion we write these functions in the form $\{p \mapsto true, ...\}$ rather than $\{(p, true), ...\}$. It is also useful to define, for some $\alpha \in 2^{Act}$ and $\omega \in \{\mathcal{L}(s) \mid s \in S\}$, the transformations $\alpha' = \{a \mapsto true \mid a \in \alpha\} \cup \{a \mapsto false \mid a \notin \alpha\}$ and $\omega' = \{p \mid p \mapsto true \in \omega\}$.

Paths are sequences of transitions where the final state of one transition equals the initial state of the next transition, if there is one. They are ranged over by σ, σ' and σ''. For a KS, $\sigma = (s_0, s_1)(s_1, s_2)...$ whereas $\sigma = (s_0, \alpha_0, s_1)(s_1, \alpha_1, s_2)...$ for a KTS or LTS. Maximal paths are either infinite or their last state has no outgoing transitions. For the set of maximal paths starting at state s we write $\mu path(s)$. For the initial and final states of the first transition of a path σ we write ${}_{\mathsf{S}}(\sigma)$ and $(\sigma)_{\mathsf{S}}$, respectively. For a KTS or LTS, we write $(\sigma)_{\mathsf{T}}$ for the set of actions of the first transition of a path σ. Usually we abbreviate these with ${}_{\mathsf{S}}\sigma$, σ_{S} and σ_{T}, respectively. A suffix σ' of a path σ is such that $\sigma = \sigma''\sigma'$ for some possibly zero length path σ''. A proper suffix σ' of a path σ is such that $\sigma = \sigma''\sigma'$ for some non-zero length path σ''. We write $\sigma \leqslant \sigma'$ when σ' is a suffix of σ and $\sigma < \sigma'$ when σ' is a proper suffix of σ.

The part time logics ACTL⁻ and UCTL⁻ are introduced. Their path formulae will be redefined later in this section to form the logics ACTL and UCTL, respectively. In what follows ϕ and ϕ' are state formulae, π and π' are path formulae.

Definition 2.4. *The syntaxes of the logics CTL, CTL*, ACTL⁻, ACTL*, UCTL⁻ and UCTL* are, with pleasing symmetry:*

CTL

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \neg\pi \mid X\phi \mid \phi\, U\phi' \mid \phi\, W\phi'$$

*CTL**

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \phi \mid \neg\pi \mid \pi \wedge \pi' \mid X\pi \mid \pi\, U\pi'$$

ACTL⁻

$$\phi ::= true \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \neg\pi \mid X\phi \mid X_a\phi \mid \phi\, U\phi' \mid \phi\, W\phi'$$

*ACTL**

$$\phi ::= true \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \phi \mid \neg\pi \mid \pi \wedge \pi' \mid X\pi \mid X_a\pi \mid \pi\, U\pi'$$

UCTL⁻

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \neg\pi \mid X\phi \mid X_a\phi \mid \phi\, U\phi' \mid \phi\, W\phi'$$

*UCTL**

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi' \mid \exists\pi$$
$$\pi ::= \phi \mid \neg\pi \mid \pi \wedge \pi' \mid X\pi \mid X_a\pi \mid \pi\, U\pi'$$

Note that for CTL, ACTL⁻, and UCTL⁻, only ϕ contributes to the formulae. For CTL*, ACTL* and UCTL*, both ϕ and π contribute to the formulae.

Definition 2.5. *The semantics of CTL and CTL* are defined over Kripke structures, KS; ACTL and ACTL* over labelled transition systems, LTS; and UCTL and UCTL* over Kripke transition systems, KTS. Specifically:*

For CTL, CTL, ACTL, ACTL*, UCTL and UCTL*:*

- $s \models \neg\phi$ *iff* $s \not\models \phi$,
- $s \models \phi \wedge \phi'$ *iff* $s \models \phi$ *and* $s \models \phi'$,
- $s \models \exists\pi$ *iff* $\exists\sigma \in \mu path(s) : \sigma \models \pi$.

For CTL, CTL, UCTL- and UCTL*:*

- $s \models p$ *iff* $\mathcal{L}(s)(p) = true$.

For CTL, CTL, ACTL, ACTL*, UCTL and UCTL*:*

- $\sigma \models \neg\pi$ *iff* $\sigma \not\models \pi$.

For CTL, ACTL- and UCTL-:

- $\sigma \models X\phi$ *iff* $\sigma_{\mathsf{S}} \models \phi$.

For ACTL- and UCTL-:

- $\sigma \models X_a\phi$ *iff* $\sigma_{\mathsf{S}} \models \phi$ *and* $a \in \sigma_{\mathsf{T}}$.

For CTL:*

- $\sigma \models X\pi$ *iff* $\exists s, s', \sigma'' : \sigma = (s, s')\sigma'', \sigma'' \models \pi$.

For ACTL and UCTL*:*

- $\sigma \models X\pi$ *iff* $\exists(s, \alpha, s')\sigma'' : \sigma = (s, \alpha, s')\sigma'', \sigma'' \models \pi$.
- $\sigma \models X_a\pi$ *iff* $\exists(s, \alpha, s')\sigma'' : a \in \alpha, \sigma = (s, \alpha, s')\sigma'', \sigma'' \models \pi$.

For CTL, ACTL* and UCTL*:*

- $\sigma \models \phi$ *iff* ${}_{\mathsf{S}}\sigma \models \phi$,
- $\sigma \models \pi \wedge \pi'$ *iff* $\sigma \models \pi$ *and* $\sigma' \models \pi'$,
- $\sigma \models \pi\, U\pi'$ *iff* $\exists\sigma' \geqslant \sigma : \sigma' \models \pi', \forall\sigma'' : \sigma \leqslant \sigma'' < \sigma' : \sigma'' \models \pi$.

For CTL, ACTL- and UCTL-:

- $\sigma \models \phi\, U\phi'$ *iff* $\exists\sigma' \geqslant \sigma :\ {}_{\mathsf{S}}\sigma' \models \phi', \forall\sigma'' : \sigma \leqslant \sigma'' < \sigma' :\ {}_{\mathsf{S}}\sigma'' \models \phi$
- $\sigma \models \phi\, W\phi'$ *iff* $\sigma \models \phi\, U\phi'$ *or* $\forall\sigma'' \geqslant \sigma :\ {}_{\mathsf{S}}\sigma'' \models \phi$.

The $\forall$ operator is defined in the usual fashion as $\forall\pi = \neg\exists\neg\pi$ where appropriate.

Definition 2.6. *Action formulae have the following syntax:*

$$\chi \;:= \tau \mid a \mid \neg\chi \mid \chi \wedge \chi'$$

Definition 2.7. *Action formulae have the following semantics:*

- $\alpha \models \tau$ *iff* $\alpha = \{\}$,
- $\alpha \models a$ *iff* $a \in \alpha$,
- $\alpha \models \neg\chi$ *iff* $\alpha \not\models \chi$,
- $\alpha \models \chi \wedge \chi'$ *iff* $\alpha \models \chi$ *and* $\alpha \models \chi'$.

For ACTL* and UCTL* we derive the following operators:

$$X_\chi\pi = \bigvee\{\bigwedge_{a\in\alpha} X_a\pi \mid \alpha \in 2^{Act}, \alpha \models \chi\}$$

$$\pi\,{}_\chi U_{\chi'}\pi' = (\pi \wedge X_\chi true)\, U\, (\pi \wedge X_{\chi'}\pi')$$

$$\pi\,{}_\chi W_{\chi'}\pi' = (\pi\,{}_\chi U_{\chi'}\pi') \vee (\pi\,{}_\chi U false)$$

These definitions cannot be applied to ACTL and UCTL. Instead we define the syntax and semantics of the logics to include them.

Definition 2.8. *The syntax of the path formulae for ACTL and UCTL is:*

$$\pi ::= X_\chi \phi \mid \phi\,_\chi U_{\chi'} \phi' \mid \phi\,_\chi W_{\chi'} \phi'$$

Definition 2.9. *The semantics of the path formulae for ACTL and UCTL are:*

- $\sigma \models X_\chi \phi$ *iff* $\sigma = \sigma'\sigma''$ *with* $\sigma'_{\mathsf{T}} \models \chi$ *and* $\sigma'_{\mathsf{S}} \models \phi$.
- $\sigma \models \phi\,_\chi U_{\chi'} \phi'$ *iff* $\exists \sigma' > \sigma : {}_{\mathsf{S}}\sigma' \models \phi, \sigma'_{\mathsf{T}} \models \chi', \sigma'_{\mathsf{S}} \models \phi', \forall \sigma \leqslant \sigma'' < \sigma' : {}_{\mathsf{S}}\sigma'' \models \phi, \sigma''_{\mathsf{T}} \models \chi$.
- $\sigma \models \phi\,_\chi W_{\chi'} \phi'$ *iff* $\sigma \models \phi\,_\chi U_{\chi'} \phi'$ *or* $\forall \sigma'' \geqslant \sigma : {}_{\mathsf{S}}\sigma'' \models \phi, \sigma''_{\mathsf{T}} \models \chi$.

Note that the W operator did not make it into the definition of ACTL in [2]. We introduce it here to bring the definition of ACTL into line with that of UCTL.

3 Mappings for 2-valued logics

We define mappings between ACTL* and CTL* in both directions and show some details of the proofs that they preserve truth. We then devise similar mappings from UCTL* to CTL* and ACTL*. We define mappings between ACTL and CTL in both directions and again show some details of the proofs that they preserve truth. And again we devise similar mappings from UCTL to CTL and ACTL.

In what follows, F is a fresh atomic proposition. Where convenient, for some ω' we write $\mathcal{L}(s)'$ where $\omega = \mathcal{L}(s)$. For convenience we also define $\{\mathsf{F}\}' = \{\mathsf{F} \mapsto true\} \cup \{p \mapsto false \mid p \in AP\}$ and $\{\mathsf{F}\}' \cup \omega' = \{\mathsf{F} \mapsto true\} \cup \omega'$.

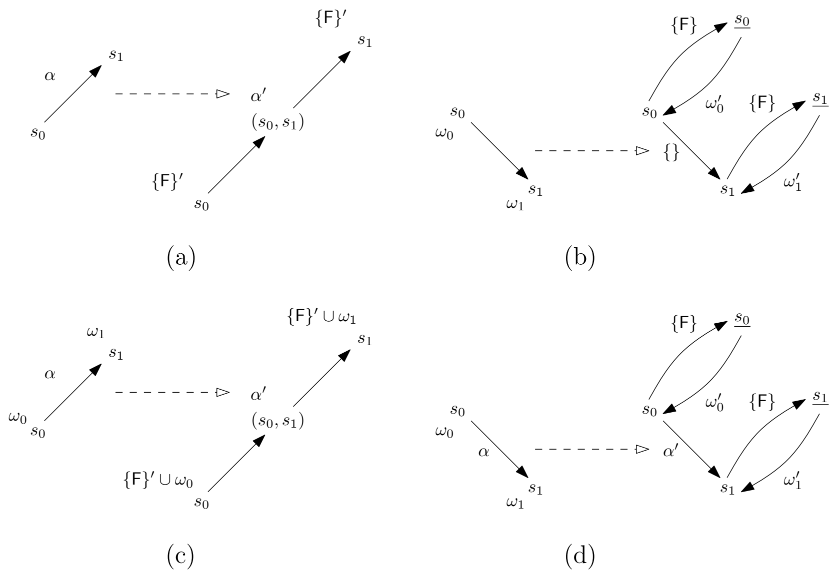

Figure 1: Mappings for 2-valued logics

3.1 ks, a mapping from ACTL* to CTL*

Let $(S, Act, \longrightarrow)$ be an LTS. The KS $(S', \longrightarrow', AP', \mathcal{L}')$ is defined:

- $S' = S \cup \{(s_0, s_1) \mid (s_0, s_1) \in \longrightarrow\}$,
- $AP' = Act \cup \{\mathsf{F}\}$,
- $\forall (s_0, \alpha, s_1) \in \longrightarrow : (s_0, (s_0, s_1)) \in \longrightarrow'$ and $((s_0, s_1), s_1) \in \longrightarrow'$,
- $\forall s \in S : \mathcal{L}'(s) = \{\mathsf{F}\}'$,
- $\forall (s_0, \alpha, s_1) \in \longrightarrow : \mathcal{L}'((s_0, s_1)) = \alpha'$.

$$\begin{aligned} ks(true) &= true \\ ks(\neg\phi) &= \neg ks(\phi) \\ ks(\phi \wedge \phi') &= ks(\phi) \wedge ks(\phi') \\ ks(\exists\pi) &= \exists ks(\pi) \end{aligned} \qquad \begin{aligned} ks(\neg\pi) &= \neg ks(\pi) \\ ks(\pi \wedge \pi') &= ks(\pi) \wedge ks(\pi') \\ ks(X\pi) &= XX(ks(\pi)) \\ ks(X_a\pi) &= Xa \wedge XX(ks(\pi)) \\ ks(\pi\, U \pi') &= (\mathsf{F} \Rightarrow ks(\pi)) U (\mathsf{F} \wedge ks(\pi')) \end{aligned}$$

Figure 1(a) shows the construction.

Theorem 3.1. *Let L be an LTS with σ a path in L and ϕ an ACTL* formula, then the mapping ks preserves truth, that is $L, \sigma \models \phi$ if and only if $ks(L), ks(\sigma) \models ks(\phi)$.*

Proof. The proof is by induction on the length of the formula. Suppose ϕ, ϕ', π, π' are all ACTL* formulae. We use the abbreviations $L_{ks} = ks(L)$, $\phi_{ks} = ks(\phi)$ and so on. The proof that $L, \sigma \models \pi\ U \pi'$ if and only if $L_{ks}, \sigma_{ks} \models (\mathsf{F} \Rightarrow \pi_{ks}) U (\mathsf{F} \wedge \pi'_{ks})$ is given. By definition, $\exists \sigma' \geqslant \sigma : L, \sigma' \models \pi', \forall \sigma \leqslant \sigma'' \leqslant \sigma' : L, \sigma'' \models \pi$. By the induction hypothesis, $L, \sigma' \models \pi'$ if and only if $L_{ks}, \sigma'_{ks} \models \pi'_{ks}$ and additionally, since $L_{ks}, \sigma'_{ks} \models \mathsf{F}$ by construction $L, \sigma' \models \pi'$ if and only if $L_{ks}, \sigma'_{ks} \models \mathsf{F} \wedge \pi'_{ks}$. Now consider $\sigma_{ks} \leqslant \sigma''_{ks} < \sigma_{ks}$. For those σ''_{ks} with $\sigma''_{ks} = (\sigma'')_{ks}$ we have $L, \sigma'' \models \pi$ if and only if $L_{ks}, \sigma''_{ks} \models \pi_{ks}$ by the induction hypothesis and again by construction, $\sigma''_{ks} \models \mathsf{F}$, hence $L, \sigma'' \models \pi$ if and only if $L_{ks}, \sigma''_{ks} \models \mathsf{F} \Rightarrow \pi_{ks}$. For those σ''_{ks} without, by construction $L_{ks}, \sigma''_{ks} \not\models \mathsf{F}$ and hence vacuously $L_{ks}, \sigma''_{ks} \models \mathsf{F} \Rightarrow \pi_{ks}$. Hence $\forall \sigma \leqslant \sigma'' < \sigma' : L, \sigma''$ if and only if $\forall \sigma_{ks} \leqslant \sigma''_{ks} < \sigma'_{ks} : L_{ks}, \sigma''_{ks} \models \mathsf{F} \Rightarrow \pi_{ks}$, which completes the given part of the proof. □

3.2 lts, a mapping from CTL* to ACTL*

Let $(S, \longrightarrow, AP, \mathcal{L})$ be a KS. The LTS $(S', Act', \longrightarrow')$ is defined:

- $S = S \cup \{\underline{s} | s \in S\}$,
- $Act' = AP \cup \{\mathsf{F}\}$,
- $\longrightarrow' = \{(s_0, \{\}, s_1) \mid (s_0, s_1) \in \longrightarrow\} \cup \{(s, \{\mathsf{F}\}, \underline{s}) \mid s \in S\} \cup \{(\underline{s}, \{\mathcal{L}(s)'\}, s) \mid s \in S\}$

$$
\begin{aligned}
lts(p) &= \exists(X_{\mathsf{F}} X_p true) & lts(\neg\pi) &= \neg lts(\pi) \\
lts(\neg\phi) &= \neg lts(\phi) & lts(\pi \wedge \pi') &= lts(\pi) \wedge lts(\pi') \\
lts(\phi \wedge \phi') &= lts(\phi) \wedge lts(\phi') & lts(X\pi) &= X(\exists X_{\mathsf{F}} true \wedge lts(\pi)) \\
lts(\exists\pi) &= \exists lts(\pi) & lts(\pi\ U \pi') &= (\exists X_{\mathsf{F}} true \wedge lts(\pi)) U (\exists X_{\mathsf{F}} true \wedge lts(\pi'))
\end{aligned}
$$

Figure 1(b) shows the construction.

Theorem 3.2. *Let K be a KS with σ a path in K and ϕ a CTL* formula, then the mapping lts preserves truth, that is $K, \sigma \models \phi$ if and only if $lts(K), lts(\sigma) \models lts(\phi)$.*

Proof. The proof is by induction on the length of the formula. Let K be a KS, then $K_{lts} = lts(K)$. The proof that $K, s \models p$ if and only if $K_{lts}, s \models \exists(X_{\mathsf{F}} X_p true)$ is given. Suppose $K, s \models p$. By construction there are transitions $(s, \{\mathsf{F}\}, s')$ and (s', ω', s) in K_{lts} with $p \in \omega'$, therefore $K_{lts}, s \models \exists(X_{\mathsf{F}} X_p true)$. Conversely, suppose $K_{lts}, s \models \exists(X_{\mathsf{F}} X_p true)$. Then there must be transitions (s, ω'_1, s_1) and (s_1, ω'_2, s_2) with $\mathsf{F} \in \omega'_1$ and $p \in \omega'_2$. By construction $\mathsf{F} \in \omega'_1$ implies both $\omega'_1 = \{\mathsf{F}\}$ and $s_1 = s'$, however. Similarly by construction $s_2 = s$ and since $\mathsf{F} \notin Act$, it can only be that this part of K_{lts} corresponds to the state $s \in K$ with $s \models p$. □

3.3 ks_2, a mapping from UCTL* to CTL*

Let $(S, Act, \longrightarrow, AP, \mathcal{L})$ be a KTS. The KS $(S', \longrightarrow', AP', \mathcal{L}')$ is defined:

- $S' = S \cup \{(s_0, s_1) \mid (s_0, s_1) \in \longrightarrow\}$,
- $AP' = AP \cup Act \cup \{\mathsf{F}\}$,
- $\forall (s_0, \alpha, s_1) \in \longrightarrow : (s_0, (s_0, s_1)) \in \longrightarrow'$ and $((s_0, s_1), s_1) \in \longrightarrow'$,
- $\forall s \in S : \mathcal{L}'(s) = \{\mathsf{F}\} \cup \mathcal{L}(s)'$,
- $\forall (s_0, \alpha, s_1) \in \longrightarrow : \mathcal{L}'((s_0, s_1)) = \alpha'$.

$$
\begin{aligned}
ks_2(p) &= p & ks_2(\neg\pi) &= \neg ks_2(\pi) \\
ks_2(\neg\phi) &= \neg ks_2(\phi) & ks_2(\pi \wedge \pi') &= ks_2(\pi) \wedge ks_2(\pi') \\
ks_2(\phi \wedge \phi') &= ks_2(\phi) \wedge ks_2(\phi') & ks_2(X\pi) &= XX(ks_2(\pi)) \\
ks_2(\exists\pi) &= \exists ks_2(\pi) & ks_2(X_a\pi) &= Xa \wedge XX(ks_2(\pi)) \\
& & ks_2(\pi\ U \pi') &= (\mathsf{F} \Rightarrow ks_2(\pi)) U (\mathsf{F} \wedge ks_2(\pi'))
\end{aligned}
$$

Figure 1(c) shows the construction.

Theorem 3.3. *Let K be a KTS with σ a path in K and ϕ a UCTL* formula, then the mapping ks_2 preserves truth, that is $K, \sigma \models \phi$ if and only if $ks_2(K), ks_2(\sigma) \models ks_2(\phi)$.* □

3.4 lts_2, a mapping from UCTL* to ACTL*

Let $(S, Act, \longrightarrow, AP, \mathcal{L})$ be a KTS. The LTS $(S', Act', \longrightarrow')$ is defined:

- $S = S \cup \{\underline{s} | s \in S\}$,
- $Act' = Act \cup AP \cup \{\mathsf{F}\}$,
- $\longrightarrow' = \{(s_0, \alpha, s_1) \mid (s_0, \alpha, s_1) \in \longrightarrow\} \cup \{(s, \{\mathsf{F}\}, \underline{s}) \mid s \in S\} \cup \{(\underline{s}, \{\mathcal{L}(s)'\}, s) \mid s \in S\}$

$$\begin{aligned} lts_2(p) &= \exists(X_{\mathsf{F}} X_p true) & lts_2(\neg\pi) &= \neg lts_2(\pi) \\ lts_2(\neg\phi) &= \neg lts_2(\phi) & lts_2(\pi \wedge \pi') &= lts_2(\pi) \wedge lts_2(\pi') \\ lts_2(\phi \wedge \phi') &= lts_2(\phi) \wedge lts_2(\phi') & lts_2(X\pi) &= X(\exists X_{\mathsf{F}} true \wedge lts_2(\pi)) \\ lts_2(\exists\pi) &= \exists lts_2(\pi) & lts_2(X_a\pi) &= X_a(lts_2(\pi)) \\ & & lts_2(\pi\, U \pi') &= (\exists X_{\mathsf{F}} true \wedge lts_2(\pi)) U (\exists X_{\mathsf{F}} true \wedge lts_2(\pi')) \end{aligned}$$

Figure 1(d) shows the construction.

Theorem 3.4. *Let K be a KTS with σ a path in K and ϕ a UCTL* formula, then the mapping lts_2 preserves truth, that is $K, \sigma \models \phi$ if and only if $lts_2(K), lts_2(\sigma) \models lts_2(\phi)$.* □

3.5 ks', a mapping from ACTL to CTL

The mapping of structures is identical to the ks mapping.

$$\begin{aligned} ks'(true) &= true \\ ks'(\neg\phi) &= \neg ks'(\phi) \\ ks'(\phi \wedge \phi') &= ks'(\phi) \wedge ks(\phi') \\ ks'(\exists\pi) &= \exists ks'(\pi) \\ ks'(\neg\pi) &= \neg ks'(\pi) \\ ks'(X_\chi \phi) &= X(\neg\mathsf{F} \wedge \chi \wedge \exists X(\mathsf{F} \wedge ks'(\phi))) \\ ks'(\phi\, {}_\chi U_{\chi'} \phi') &= ((\mathsf{F} \wedge ks'(\phi)) \vee (\neg\mathsf{F} \wedge \chi)) U (\neg\mathsf{F} \wedge \exists((\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge ks'(\phi')))) \\ ks'(\phi\, {}_\chi W_{\chi'} \phi') &= ((\mathsf{F} \wedge ks'(\phi)) \vee (\neg\mathsf{F} \wedge \chi)) W (\neg\mathsf{F} \wedge \exists((\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge ks'(\phi')))) \end{aligned}$$

Theorem 3.5. *Let L be an LTS with σ a path in L and ϕ an ACTL formula, then the mapping ks' preserves truth, that is $L, \sigma \models \phi$ if and only if $ks'(L), ks'(\sigma) \models ks'(\phi)$.*

Proof. The proof is by induction on the length of the formula. Let L be an LTS with ϕ, ϕ', π, π' formulae satisfied in L. Then $L_{ks'} = ks'(L)$, $\phi_{ks'} = ks'(\phi)$ and so on. The proof that $L, \sigma \models \phi_\chi U_{\chi'} \phi'$ if and only if $L_{ks'}, \sigma_{ks'} \models ((\mathsf{F} \wedge \phi_{ks'}) \vee (\neg\mathsf{F} \wedge \chi)) U (\neg\mathsf{F} \wedge \exists((\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge \phi'_{ks'})))$ is given. By definition, $L, \sigma \models \phi_\chi U_{\chi'} \phi'$ if and only if $\exists\sigma' > \sigma : L, {}_{\mathsf{S}}\sigma' \models \phi, L, \sigma'_{\mathsf{T}} \models \chi', L, \sigma'_{\mathsf{S}} \models \phi', \forall\sigma \leqslant \sigma'' < \sigma' : L, {}_{\mathsf{S}}\sigma'' \models \phi, L, \sigma''_{\mathsf{T}} \models \chi$. Consider ${}_{\mathsf{S}}\sigma'$. By the induction hypothesis $L, {}_{\mathsf{S}}\sigma' \models \phi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma'_{ks'}) \models \phi_{ks'}$ and by construction ${}_{\mathsf{S}}(\sigma'_{ks'}) \models \mathsf{F}$, therefore $L, {}_{\mathsf{S}}\sigma' \models \phi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma'_{ks'}) \models \mathsf{F} \wedge \phi_{ks'}$. Now consider σ'_{T} and σ'_{S}. By construction $L, \sigma'_{\mathsf{T}} \models \chi'$ if and only if $L_{ks'}, (\sigma'_{ks'})_{\mathsf{S}} \models \chi'$. Also, by the induction hypothesis $L, \sigma'_{\mathsf{S}} \models \phi'$ if and only if $L_{ks'}, (\sigma'_{\mathsf{S}})_{ks'} \models \phi'_{ks'}$ and by construction $L_{ks'}, (\sigma'_{\mathsf{S}})_{ks'} \models \mathsf{F}$ therefore $L, \sigma'_{\mathsf{S}} \models \phi'$ if and only if $L_{ks'}, (\sigma'_{\mathsf{S}})_{ks'} \models \mathsf{F} \wedge \phi'_{ks'}$. Working in $L_{ks'}$, it remains to be shown that $(\sigma'_{ks'})_{\mathsf{S}} \models \chi'$ and $(\sigma'_{\mathsf{S}})_{ks'} \models \mathsf{F} \wedge \phi'_{ks'}$ if and only if ${}_{\mathsf{S}}((\sigma'_{ks'})_{\mathsf{S}}, (\sigma'_{\mathsf{S}})_{ks'}) \models \neg\mathsf{F} \wedge \exists((\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge \phi'_{ks'}))$. From left to right we have $(\sigma'_{ks'})_{\mathsf{S}} \models \neg\mathsf{F}$ hence ${}_{\mathsf{S}}((\sigma'_{ks'})_{\mathsf{S}}, (\sigma'_{\mathsf{S}})_{ks'}) \models \neg\mathsf{F}$. Similarly $(\sigma'_{ks'})_{\mathsf{S}} \models \neg\mathsf{F} \wedge \chi'$ and $(\sigma'_{ks'})_{\mathsf{S}} \models \mathsf{F} \wedge \phi'_{ks'}$ hence ${}_{\mathsf{S}}((\sigma'_{ks'})_{\mathsf{S}}, (\sigma'_{\mathsf{S}})_{ks'}) \models (\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge \phi'_{ks'})$. The result then follows. From right to left the argument is similar. Therefore $\sigma'_{\mathsf{T}} \models \chi'$ and $\sigma'_{\mathsf{S}} \models \phi$ if and only if ${}_{\mathsf{S}}((\sigma'_{ks'})_{\mathsf{S}}, (\sigma'_{\mathsf{S}})_{ks'}) \models \neg\mathsf{F} \wedge \exists((\neg\mathsf{F} \wedge \chi') U (\mathsf{F} \wedge \phi'_{ks'}))$. Lastly consider ${}_{\mathsf{S}}\sigma''$ and σ''_{T} with $\sigma_{ks'} \leqslant \sigma''_{ks'} < \sigma'_{ks'}$. For those $\sigma''_{ks'}$ with $\sigma''_{ks'} = (\sigma'')_{ks'}$ we have $L, {}_{\mathsf{S}}(\sigma'') \models \phi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \models \phi_{ks'}$ and by construction $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \models \mathsf{F}$. Therefore $L, {}_{\mathsf{S}}\sigma'' \models \phi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \models \mathsf{F} \wedge \phi_{ks'}$. For each of those $\sigma''_{ks'}$ without, by construction there is a unique σ'' with $L, \sigma''_{\mathsf{T}} \models \chi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \models \chi$. Also by construction $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \not\models \mathsf{F}$ and therefore $L, \sigma''_{\mathsf{T}} \models \chi$ if and only if $L_{ks'}, {}_{\mathsf{S}}(\sigma''_{ks'}) \models \neg\mathsf{F} \wedge \chi$. Hence $\forall\sigma \leqslant \sigma'' < \sigma' : {}_{\mathsf{S}}\sigma'' \models \phi, \sigma''_{\mathsf{T}} \models \chi$ if and only if $\forall\sigma_{ks'} \leqslant \sigma''_{ks'} < \sigma'_{ks'} : \sigma''_{ks'} \models (\mathsf{F} \wedge \phi_{ks'}) \vee (\neg\mathsf{F} \wedge \chi)$, which completes the given part of the proof. □

3.6 lts', a mapping from CTL to ACTL

The mapping of structures is identical to the lts mapping.

$$
\begin{aligned}
lts'(p) &= \exists X_{\mathsf{F}}(\exists X_p true) & lts'(\neg\pi) &= \neg lts'(\pi)\\
lts'(\neg\phi) &= \neg lts'(\phi) & lts'(X\phi) &= X(lts'(\phi))\\
lts'(\phi\wedge\phi') &= lts'(\phi)\wedge lts'(\phi') & lts'(\phi\, U\phi') &= (\exists X_{\mathsf{F}} true \wedge lts'(\phi))U(\exists X_{\mathsf{F}} true\wedge lts'(\phi'))\\
lts'(\exists\pi) &= \exists lts'(\pi) & lts'(\phi\, W\phi') &= (\exists X_{\mathsf{F}} true \wedge lts'(\phi))W(\exists X_{\mathsf{F}} true\wedge lts'(\phi'))
\end{aligned}
$$

Theorem 3.6. *Let K be a KS with σ a path in K and ϕ a CTL formula, then the mapping lts' preserves truth, that is $K,\sigma\models\phi$ if and only if $lts'(K), lts'(\sigma)\models lts'(\phi)$.* □

3.7 ks_2', a mapping from UCTL to CTL

The mapping of structures is identical to the ks_2 mapping.

$$
\begin{aligned}
ks_2'(p) &= p\\
ks_2'(\neg\phi) &= \neg ks_2'(\phi)\\
ks_2'(\phi\wedge\phi') &= ks_2'(\phi)\wedge ks_2(\phi')\\
ks_2'(\exists\pi) &= \exists ks_2'(\pi)\\
ks_2'(\neg\pi) &= \neg ks_2'(\pi)\\
ks_2'(X_\chi\phi) &= X(\neg\mathsf{F}\wedge\chi\wedge\exists X(\mathsf{F}\wedge ks_2'(\phi')))\\
ks_2'(\phi\,{}_\chi U_{\chi'}\phi') &= ((\mathsf{F}\wedge ks_2'(\phi))\vee(\neg\mathsf{F}\wedge\chi))U(\neg\mathsf{F}\wedge\exists((\neg\mathsf{F}\wedge\chi')U(\mathsf{F}\wedge ks_2'(\phi'))))\\
ks_2'(\phi\,{}_\chi W_{\chi'}\phi') &= ((\mathsf{F}\wedge ks_2'(\phi))\vee(\neg\mathsf{F}\wedge\chi))W(\neg\mathsf{F}\wedge\exists((\neg\mathsf{F}\wedge\chi')U(\mathsf{F}\wedge ks_2'(\phi'))))
\end{aligned}
$$

Theorem 3.7. *Let K be a KTS with σ a path in K and ϕ a UCTL formula, then the mapping ks_2' preserves truth, that is $K,\sigma\models\phi$ if and only if $ks_2'(K), ks_2'(\sigma)\models ks_2'(\phi)$.* □

3.8 lts_2', a mapping from UCTL to ACTL

The mapping of structures is identical to the lts_2' mapping.

$$
\begin{aligned}
lts_2'(p) &= \exists X_{\mathsf{F}}(\exists X_p true)\\
lts_2'(\neg\phi) &= \neg lts_2'(\phi)\\
lts_2'(\phi\wedge\phi') &= lts_2'(\phi)\wedge lts_2'(\phi')\\
lts_2'(\exists\pi) &= \exists lts_2'(\pi)\\
lts_2'(\neg\pi) &= \neg lts_2'(\pi)\\
lts_2'(X_\chi\pi) &= X_\chi(\exists X_{\mathsf{F}} true\wedge lts_2'(\pi))\\
lts_2'(\phi\,{}_\chi U_{\chi'}\phi') &= (\exists X_{\mathsf{F}} true\wedge lts_2'(\phi))\,{}_\chi U_{\chi'}(\exists X_{\mathsf{F}} true\wedge lts_2'(\phi'))\\
lts_2'(\phi\,{}_\chi W_{\chi'}\phi') &= (\exists X_{\mathsf{F}} true\wedge lts_2'(\phi))\,{}_\chi W_{\chi'}(\exists X_{\mathsf{F}} true\wedge lts_2'(\phi'))
\end{aligned}
$$

Theorem 3.8. *Let K be a KTS with σ a path in K and ϕ a UCTL formula, then the mapping lts_2' preserves truth, that is $K,\sigma\models\phi$ if and only if $lts_2'(K), lts_2'(\sigma)\models lts_2'(\phi)$.* □

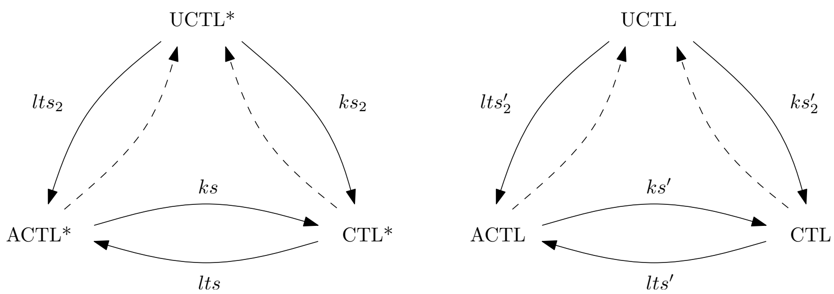

Figure 2: A summary of the mappings for 2-valued logics

4 Definitions for 3-valued logics

We define a variant of Kripke modal transition systems with *must* and *may* transitions replaced by modifiers on the actions, some common concepts, and then the syntax and semantics of the logic UPML. Note that in the definitions which follow we again limit the number of transitions between any two states in any one direction to at most one.

Definition 4.1. *For a set of actions Act, a set of modified actions Act_M is defined as $Act_M \subseteq Act \times \{!, ?\}$. We write the elements of Act_M in shorthand notation, for example $(a, !)$ becomes $a!$ and so on. For any set Act_M we place restrictions on the elements, namely $a! \in Act_M \Rightarrow a? \notin Act_M$ or, equivalently, $a? \in Act_M \Rightarrow a! \notin Act_M$.*

Definition 4.2. *A Kripke modal transition system or KMTS is a tuple $(S, Act_M, \longrightarrow, AP, \mathcal{L})$ where:*

- *S is a set of states ranged over by $s, s_0, s_1, ...,$*
- *Act_M is set of modified actions ranged over by $a?, a!$ with $\alpha, \alpha_0, \alpha_1, ...$ ranging over 2^{Act_M},*
- *$\longrightarrow \subseteq S \times S$ is the transition relation with $(s_0, s_1) \in \longrightarrow$,*
- *AP is a set of atomic propositions ranged over by p,*
- *$\mathcal{L} : S \times AP \longrightarrow \{true, \perp, false\}$ is an interpretation function that associates a value of $true$, $false$ or $\perp$, meaning unknown, with each $p \in AP$ for each $s \in S$,*
- *For any two transitions, $(s_0, \alpha_0, s_1), (s_0, \alpha_1, s_1) \in \longrightarrow \Rightarrow \alpha_0 = \alpha_1$.*

We overload $\mathcal{L}$ and define, for each $s \in S$, the function $\mathcal{L}(s) : AP \longrightarrow \{true, \perp, false\}$ where $\mathcal{L}(s)(p) = \mathcal{L}(s, p)$. We also define, for $\alpha \in 2^{Act_M}$ and $\omega \in \{\mathcal{L}(s) \mid s \in S\}$, the transformations $\alpha' = \{a \mapsto true \mid a! \in \alpha\} \cup \{a \mapsto \perp \mid a? \in \alpha\} \cup \{a \mapsto false \mid a! \notin \alpha \wedge a? \notin \alpha\}$ and $\omega' = \{p! \mid p \mapsto true \in \omega\} \cup \{p? \mid p \mapsto \perp \in \omega\}$.

Paths and their related definitions are defined in an entirely analogous fashion to those for 2-valued structures.

To interpret propositional operators we use Kleene's strong 3-valued propositional logic [8]. Negation maps $true$ to $false$, $false$ to $true$ and $\perp$ to $\perp$. The 3-valued truth table for conjunction and disjunction, the latter derived from the former by way of De Morgan's law $x \vee y = \neg(\neg x \wedge \neg y)$, is given below.

x	y	$x \wedge y$	$x \vee y$	x	y	$x \wedge y$	$x \vee y$	x	y	$x \wedge y$	$x \vee y$
f	f	f	f	$\perp$	f	f	$\perp$	t	f	f	t
f	$\perp$	f	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	t	$\perp$	$\perp$	t
f	t	f	t	$\perp$	t	$\perp$	t	t	t	t	t

The logic UPML is introduced, which has the characteristics of both 3-valued PML [1] and 3-valued $\text{PML}^{\mathtt{Act}}$.

Definition 4.3. *The syntax the logic UPML is:*

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi' \mid AX\phi \mid AX_a\phi$$

Definition 4.4. *The semantics of UPML are defined over Kripke modal transition sytems, KMTS. Specifically:*

$$[s \models \neg\phi] = \neg\, [s \models \phi]$$

$$[s \models \phi \wedge \phi'] = [s \models \phi] \wedge [s \models \phi']$$

$$[s \models AX\phi] = \begin{cases} true & \forall (s, \_, s') : [s' \models \phi] = true \\ false & \exists (s, \_, s') : [s' \models \phi] = false \\ \perp & otherwise \end{cases}$$

$$[s \models AX_a\phi] = \begin{cases} true & \forall (s, \alpha, s') : (a! \in \alpha \vee a? \in \alpha) \Rightarrow [s' \models \phi] = true \\ false & \exists (s, \alpha, s') : \qquad a! \in \alpha \wedge [s' \models \phi] = false \\ \perp & otherwise \end{cases}$$

Here the underscore character _ stands for any set of modified actions. The operators EX and EX_a can be derived in the usual manner. We give the semantics of the latter by way of an example, however:

$$[s \models EX_a\phi] = \begin{cases} true & \exists(s,\alpha,s') : \qquad\qquad\qquad a! \in \alpha \wedge [s' \models \phi] = true \\ false & \forall(s,\alpha,s') : (a! \in \alpha \vee a? \in \alpha) \Rightarrow [s' \models \phi] = false \\ \perp & otherwise \end{cases}$$

Modifying actions with the ! and ? modifiers is equivalent to their transitions being *must* and *may* transitions, respectively. Note that this definition is a departure from convention [6, 7] and that the correspondence is not quite straightforward, since all *must* transitions are also *may* transitions whereas actions are modified with only one of the ! and ? modifiers, not both. The correspondence is effectively a bijection, however, in terms of the above definitions. In the definitions of $[s \models AX_a\phi]$ and $[s \models EX_a\phi]$, for example, we see the term $(a! \in \alpha \vee a? \in \alpha)$ rather than just $a! \in \alpha$, making them entirely consistent with those of [7].

Finally, we note that the AX_a operator defined here has a different quality to the composite $\forall X_a$ operator defined in the 2-valued case. This has nothing to do with the 3-valued nature of the these logics nor the presence of ! and ? modifiers on the actions. Specifically, in the case of 2-valued logics, the $X_a\phi$ operator is satisfied by a path that is both labelled by an action *a and* who's next state satisfies ϕ. Addition of the $\forall$ operator then ensures that all paths from a given state satisfy this condition. In the case of 3-valued logics, however, leaving aside the 3-valued nature of the logics and the modified actions, which do not affect the argument, the AX_a operator is satisfied when all paths from a state satisfy the condition that labelling by an action *a implies* that the next state satisfies ϕ.

5 Conclusions

We have defined the logic UCTL* over modified Kripke transition systems. Given recent developments [4, 5], we claim this step is a worthwhile one. We have also modified Kripke modal transition systems in similar fashion and defined a logic, UPML, over these systems. We have defined mappings between the various logics in the 2-valued case that preserve truth but have shied away from such an approach in the 3-valued case. As reported in [3], the results of [2] led to practial gains in model checking at the time but with the plethora of model checkers around today it seems unlikely that similar results for 3-valued logics will have any impact. The results of [6] are not quite complete, their Kripke modal transition systems do not carry actions on their transitions, but a full investigation of the 3-valued case is likely to have only theoretical interest and is therefore left for future work.

5.1 Acknowledgements

My sincere thanks to Ian Hodkinson for his many helpful comments and corrections.

References

[1] Glenn Bruns and Patrice Godefroid. Model checking partial state spaces with 3-valued temporal logics. In *CAV*, pages 274–287, 1999.

[2] Rocco De Nicola and Frits W. Vaandrager. Action versus state based logics for transition systems. In *Semantics of Systems of Concurrent Processes*, pages 407–419, 1990.

[3] Rocco De Nicola and Frits W. Vaandrager. Three logics for branching bisimulation (extended abstract). In *LICS*, pages 118–129, 1990.

[4] Alessandro Fantechi, Stefania Gnesi, Alessandro Lapadula, Franco Mazzanti, Rosario Pugliese, and Francesco Tiezzi. A logical verification methodology for service-oriented computing. *ACM Trans. Softw. Eng. Methodol.*, 21(3):16, 2012.

[5] Stefania Gnesi and Franco Mazzanti. An abstract, on the fly framework for the verification of service-oriented systems. In Martin Wirsing and Matthias M. Hölzl, editors, *Results of the SENSORIA Project*, volume 6582 of *Lecture Notes in Computer Science*, pages 390–407. Springer, 2011.

[6] Patrice Godefroid and Radha Jagadeesan. On the expressiveness of 3-valued models. In *VMCAI*, pages 206–222, 2003.

[7] Michael Huth, Radha Jagadeesan, and David A. Schmidt. Modal transition systems: A foundation for three-valued program analysis. In *ESOP*, pages 155–169, 2001.

[8] Stephen Cole Kleene. *Introduction to Metamathematics*. North Holland, 1952.

[9] Kim Guldstrand Larsen and Bent Thomsen. A modal process logic. In *LICS*, pages 203–210, 1988.

[10] Markus Müller-Olm, David A. Schmidt, and Bernhard Steffen. Model-checking: A tutorial introduction. In *SAS*, pages 330–354, 1999.

[11] David A. Schmidt. Binary relations for abstraction and refinement. In *in: Workshop on Refinement and Abstraction*. Elsevier Electronic, 2000.

[12] Maurice H. ter Beek, Alessandro Fantechi, Stefania Gnesi, and Franco Mazzanti. A state/event-based model-checking approach for the analysis of abstract system properties. *Sci. Comput. Program.*, 76(2):119–135, 2011.